Journal of Applied Analysis and Computation
Volume 16, Number 4, August 2026, 1805–1821
Website:http://www.jaac-online.com
DOI:10.11948/20250078

A CONCISE TRAINING SCHEME OF RBF NEURAL NETWORKS WITH FIXED CENTER POINTS FOR PARTIAL DIFFERENTIAL EQUATIONS WITH DIRICHLET BOUNDARY CONDITIONS*

Yanting Li[1], Huantian Xie[1,†], Juan Zhang[2,†] and Jianwei Zhou[1]

Abstract A concise numerical scheme based on radial basis function neural networks is proposed for solving second order partial differential equations with Dirichlet boundary conditions. By employing fixed center points across the geometric region, our approach reduces the memory requirements. And the shape parameters are obtained by machine learning, which overcomes the shortcomings of artificially selection about parameters. Specially, the corresponding numerical schemes are calculated by nonlinear least squares problems, avoiding directly solving linear algebraic equations. The proposed training scheme of RBF neural networks, which bases on Gaussian and multiquadric radial basis functions, significantly achieves high-accuracy approximations and improves the computational efficiency compared to existing meshless methods. In comparison with Kansa's method and the radial basis function neural networks method proposed in [22], numerical experiments demonstrate that our approximation schemes perform well in various domains. Through comprehensive numerical experiments comparing Kansa's method and radial basis function neural networks, our proposed approximation schemes also exhibit better performance across multiple geometric domains, such as unit square, peanut-shaped and five-petaled domains.

Keywords RBF neural network, mesh-free method, Gaussian function, multiquadric function, high accuracy approximation.

MSC(2010) 65N80, 65D12, 68TB07.

1. Introduction

Recently, mesh-free methods have gradually emerged and become some of the dominant numerical methods for solving general science and engineering problems due to their simplicity and effectiveness, see [6, 8, 22] and monograph [25] and the references cited therein. A single-step implicit block method involving one hybrid point was proposed to solve nonlinear equations in [12]. An improved collocation technique had been proposed to simulate wave behaviours of the fourth-order multi-parameter non-linear Kuramoto-Sivashinsky equation in [26]. Hybrid quintic hermite splines collocation technique was employed to approximate travelling wave solution of

†The corresponding authors.
[1]School of Mathematics and Statistics, Linyi University, Linyi 276005, China
[2]School of Logistics, Linyi University, Linyi 276005, China
*The authors were sorted according to the alphabetical sequence of their surnames, and partly supported by National Natural Science Foundation of China (12271233, 12101283), Improving innovation ability of enterprises in Shandong province (2023TSGC0466) and Entrusted research of Hunan Shaofeng Institute for Applied Mathematics (KF-2022-11-01).
Email: 26230696@qq.com(Y. Li), huantianxie@lyu.edu.cn(H. Xie),
jzhang_math@hotmail.com(J. Zhang), jwzhou@yahoo.com(J. Zhou)

fourth order Fisher Kolmogorov reaction diffusion equation in [27].

One of the primary advantages of mesh-free methods is that they don't require either intricate domain partitions or boundary discretizations, and can approximate partial differential equations (PDEs) in complicated irregular domains, models in higher dimensions, moving boundary cases and so on. Radial basis functions (RBF) are real-valued functions depending only on the distance from some selected points, $\Phi(x) = \Phi(\|x - x_n\|)$, where $\{x_n\}$ are called as centers [7, 25]. The authors constructed a fully discrete formulation based upon the RBF-finite difference method for improved Boussinesq model in [1]. Furthermore, the RBF collocation technique was mixed with a finite difference scheme to solve time-dependent PDEs in [14]. [23] presented a new high-order, local RBF Hermite finite difference meshfree method to solve the reaction-diffusion equations on smooth surfaces. The improved accuracy and computational cost are demonstrated with a series of numerical examples. To improved the accuracy in space direction, [17] used the temporal Richardson extrapolation technique and RBF generated Hermite finite difference to solve the cubic-quintic complex Ginzburg-Landau equation. Generally, the RBF centers are placed within the domain. With these center points, RBF collocation methods provide great flexibilities for choosing basis functions, and have attracted considerable attention (for details please refer to [8, 30] and the references cited therein). Without doubt, Kansa's method [21] is the best-known RBF collocation method, which employs the same centers as collocation points. However, Chen stated that extending the centers beyond the physical domain leads to improved performance for collocation methods [6]. In view of mesh-free methods, this idea is analogous to the use of artificial source points located outside the domain by the method of fundamental solutions [9,11,15]. Meanwhile, this numerical scheme has a considerable impact on the accuracy without any additional computational cost.

A neural network, commonly known as an artificial neural network, is a biologically inspired adaptive system that processes information through interconnected neurons structured in multiple layers, analogous to the human brain's learning mechanism. And the neural networks techniques are widely used for learning and solving complex pattern recognition or function approximation problems (please refer to [10] and the references cited therein). The authors developed RBF neural networks methods for approximating boundary value problems in [16], which depend on the error functional minimization. And based on the trust region method, they also proposed a learning RBF network that significantly reduces the time needed for tuning their parameters. A mesh-free method using RBF networks with trust region was proposed to solve two-dimensional Poisson equation in [4], which simplifies the process of network structure selection and reduces time expenses. A new efficient RBF neural network scheme was proposed for solving two- and three–dimensional elliptic PDEs in [22], whose points are separated and extended to regions covering irregular domains. Meanwhile, RBF neural networks had been proposed to approximate the inverse PDEs, which were trained via the least squares minimization of a nonlinear functional in [20]. Adaptive RBF neural networks were used to simulate a class of time varying system in the presence of model uncertainties and external disturbance under the same Lyapunov framework [13]. Hybrid RBF neural network was selected as a forecaster to study the daily movements of stock indices [2]. And their results demonstrated the utility and the flexibility of Hybrid RBF neural networks as a clever predictive modeling tool for highly dependent and nonlinear data.

However, during the iterations or learning proceedings, one has to refine the centers within the hidden layer of neural networks, which consumes lots of computing resources. Combining RBF neural networks methods with fixed centers, one could design more efficient numerical network schemes, which omit the learning behavior of hidden layer weights and avoid the

time-consuming layer-by-layer transmission process. Our strategic choice simplifies the training process by fixing the center locations, eliminating the need to optimize these parameters during network learning. Consequently, it reduces the number of variables in the optimization problem, potentially accelerating convergence. Therefore, the learning convergence speed of the network will be very fast. In order to investigate the efficiency, we choose fixed centers to propose a novel mesh-free approach with neural networks techniques in this work. Specially the unknown-coefficients and shape-parameters are determined by a three-layer neural network. And numerical results are given for different regions with a better accuracy than schemes in the current literature.

The rest of the paper is organized as follows. In Section 2, we introduce two RBF and some preliminaries which will be used in the sequel. In Section 3, the formulation of the proposed neural network method for PDEs is given in detail. And the centers are selected as fixed points during the hidden layer of the neural networks. In Section 4, some numerical examples are listed to demonstrate the efficiency and advantages of our proposed approximations. Finally, some conclusions are provided in Section 5.

2. Back settings and preliminaries

The RBF neural network is an artificial neural network consisting of three layers, i.e., input, hidden and output layers. It differs from traditional feedforward neural networks in its structure and working principle, which mainly utilizes RBF in the hidden layer for information proceedings. The approximation accuracy of RBF containing shape parameters largely depends on the selection of shape parameters. Here we first recall the two well-known and widely used classic RBFs, namely, the Gaussian and the multiquadric functions (the details please refer to [22,30]).

The Gaussian function is defined as:

$$\Phi(x) = e^{-c|x|^2}, \tag{2.1}$$

where $|x|$ is the Euclidean distance from the origin to x, and c called a shape parameter describing the shape of the Gaussian function. It is clear that if $|x|$ increases, $\Phi(x)$ decreases exponentially. Similarly, with the same variable notations, the multiquadric function is defined as:

$$\Phi(x) = \sqrt{1 + (c|x|)^2}. \tag{2.2}$$

Obviously, the multiquadric function increases quasi-linearly as $|x|$ increases. We use the expression $A \lesssim B$ to mean that there exists a generic positive constant C such that $A \leq CB$.

For infinitely smooth RBF, such as the Gaussian or multiquadric functions, the pointwise error of RBF interpolation $s(x)$ for sufficiently smooth function $f(x)$ is bounded:

$$\|f - s\|_{L_\infty(\Omega)} \lesssim e^{-\frac{1}{c}},$$

where c is the shape parameter of the RBF. As $c \to 0$, the error exponentially decreases. However, this benefit is coupled with a severe drawback: The condition number of the RBF interpolation matrix grows quickly. To overcome this challenge, our scheme replace the direct solution of a linear algebraic system by a nonlinear least-squares optimization. This approach avoids the numerical instability caused by ill-conditioned matrices since it eliminates the need for explicit matrix inversion. Consequently, we can determine optimal shape parameters that maintain high accuracy while preserving numerical stability. In this framework, both the RBF

weights and shape parameters can be treated as trainable variables, which are simultaneously optimized by minimizing a given loss function.

In this paper, we focus on the two dimensional regions, however, the same techniques can be extended for high dimensional cases. Hence for all $\mathbf{x} = (x, y) \in \Omega \subset \mathbb{R}^2$, the interior centers $\{\hat{\mathbf{x}}_n\}_{n=1}^{N_{\text{int}}}$ and the boundary centers $\{\hat{\mathbf{x}}_n\}_{n=N_{\text{int}}+1}^{N_{\text{int}}+N_{\text{bry}}}$ comprise the set of centers $\{\hat{\mathbf{x}}_n\}_{n=1}^{N}$, where $N = N_{\text{int}} + N_{\text{bry}}$. Meanwhile the interior collocation points $\{\mathbf{x}_m\}_{m=1}^{M_{\text{int}}}$ and the boundary collocation points $\{\mathbf{x}_m\}_{m=M_{\text{int}}+1}^{M_{\text{int}}+M_{\text{bry}}}$ comprise the set of collocation points $\{\mathbf{x}_m\}_{m=1}^{M} \in \Omega$, where $M = M_{\text{int}} + M_{\text{bry}}$. We set the basis functions as $\{\omega(\mathbf{c}_n; r_n)\}_{n=1}^{N}$, which depend on shape parameters $\mathbf{c}_n$ and $r_n \triangleq |\mathbf{x} - \hat{\mathbf{x}}_n|$. For convenience, the collocation points are selected as a fixed distribution of Halton points either in the interior or on the boundary. Similarly, the interior centers and the boundary centers are uniformly distributed in the domain.

In the following parts, we will restrict our discussion to second-order PDEs with general boundary conditions

$$\begin{cases} \mathcal{L}u(\mathbf{x}) = f(\mathbf{x}), \quad \forall\, \mathbf{x} \in \Omega, \\ \mathcal{B}u(\mathbf{x})\big|_{\partial\Omega} = g(\mathbf{x}), \end{cases} \tag{2.3}$$

where $\mathcal{L}$ is an elliptic operator and linear operator $\mathcal{B}$ describes the boundary conditions.

We can rephrase the above PDEs as a minimization problem, which differs from the classical approximation problem in that there are unknown values of the function on a predefined set of sampling points. Following the mesh-free techniques, it is possible to minimize the residuals at sampling points inside and on the boundary of the region. Note that, for any vector function

$$\mathbf{G}(\mathbf{x}) = \begin{bmatrix} g_1(\mathbf{x}) \\ g_2(\mathbf{x}) \\ \vdots \\ g_n(\mathbf{x}) \end{bmatrix}, \tag{2.4}$$

$\mathbf{G}(\mathbf{x}) = \mathbf{0}$ is equivalent to solving

$$\min_{\mathbf{x}} \|\mathbf{G}(\mathbf{x})\|_2^2 = \min_{\mathbf{x}} \left(g_1(\mathbf{x})^2 + g_2(\mathbf{x})^2 + \ldots + g_n(\mathbf{x})^2 \right), \tag{2.5}$$

where $\mathbf{G}(\mathbf{x}) \triangleq \{\mathcal{L}u(\mathbf{x}) - f(\mathbf{x}),\ \mathcal{B}u(\mathbf{x}) - g(\mathbf{x})\}$ on $\{\mathbf{x}_n\}_{n=1}^{M}$.

3. RBF neural networks method

In this section, we give a specific numerical approach with the RBF neural networks to approximate (2.3). We use the nonlinear least squares method to train the neural networks and improve the accuracy of numerical solutions within iterations. In fact, the second layer consists of the RBF that produce a nonlinear transformation of the input vector. When solving PDEs, the input vector consists of the coordinates of the centers at which the approximations to the solution will be calculated.

The RBF neural network is a specialized type of artificial neural network, and commonly used to fit and approximate complex nonlinear relationships. Here, the solution u of (2.3) can

be approximated by

$$u_N(\mathbf{x}) = \sum_{i=1}^{N} a_i \omega(c_i; r_i(\mathbf{x})), \quad \forall \mathbf{x} \in \Omega, \tag{3.1}$$

where $r_i(\mathbf{x}) = |\mathbf{x} - \hat{\mathbf{x}}_i|$. Note that the number of center point is equivalent to the size of RBF neural networks. In order to reduce the number of network parameters and save storage, we choose and fix the locations of the center points. [18, 19, 29] pointed that the accuracy of RBF methods with variable shape parameters may have an advantage over a constant shape parameter strategy. The total set of network parameters consists of $2N$ unknown coefficients $\mathbf{a} = [a_1, a_2, \ldots, a_N]^\intercal$ and the shape parameters $\mathbf{c} = [c_1, c_2, \ldots, c_N]^\intercal$. These coefficients are calculated by training the network to satisfy the following equations at the M collocation points:

$$\begin{cases} \mathcal{L}u_N(\mathbf{x}_i) = f(\mathbf{x}_i), & i = 1, \ldots, M_{\text{int}}, \\ \mathcal{B}u_N(\mathbf{x}_i) = g(\mathbf{x}_m), & i = M_{\text{int}} + 1, \ldots, M_{\text{int}} + M_{\text{bry}}, \end{cases} \tag{3.2}$$

which yield M non-linear equations. Here we set $M \geq 2N$ to guarantee the solvability of the discretized numerical schemes.

In light of the above settings, we rewrite the numerical scheme of (2.3) as

$$\mathbf{F} = \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_M \end{bmatrix} = \begin{bmatrix} \mathcal{L}u_N(\mathbf{x}_1) - f(\mathbf{x}_1) \\ \vdots \\ \mathcal{L}u_N(\mathbf{x}_{M_{\text{int}}}) - f(\mathbf{x}_{M_{\text{int}}}) \\ \sqrt{\lambda}(\mathcal{B}u_N(\mathbf{x}_{M_{\text{int}}+1}) - g(\mathbf{x}_{M_{\text{int}}+1})) \\ \vdots \\ \sqrt{\lambda}(\mathcal{B}u_N(\mathbf{x}_M) - g(\mathbf{x}_M)) \end{bmatrix} = \mathbf{0}, \tag{3.3}$$

where λ is a fitted penalty factor accounting for the weight of the boundary conditions.

Here we are at the point to calculate the following minimization problem:

$$\min_{\mathbf{a},\mathbf{c}} \mathcal{S} = \sum_{j=1}^{M_{\text{int}}} [\mathcal{L}u_N(\mathbf{x}_j) - f(\mathbf{x}_j)]^2 + \lambda \sum_{j=M_{\text{int}}+1}^{M} [\mathcal{B}u_N(\mathbf{x}_j) - g(\mathbf{x}_j)]^2. \tag{3.4}$$

To solve the above nonlinear least squares problem, we use a trust-region reflective algorithm, which is a widely used optimization algorithm (more details please refer to [3,5] and the references cited therein). Traditional RBF methods are equipped with a single-layer structure. The input is directly transformed by the radial basis functions, and the outputs of these functions are then linearly combined to produce the final output. The RBF neural networks usually are based on three-layer feedforward neural networks, i.e., the input layer receives the input data, the hidden layer contains several RBF units (neurons), and the output layer performs a linear weighted sum of the outputs from the hidden layer RBF units to produce the network's final output. Specially, each RBF unit has a center and a shape parameter.

The RBF neural networks described above consists of three layers. The training (collocation) points $\{\mathbf{x}_j\}_{j=1}^{M}$ are input into the first layer. The corresponding RBF, which are equipped with nonlinear transformations, are calculated in the second layer to train the weights and shape

parameters. Hence, we construct the weighted sum of these RBF with a linear transformation to yield the numerical solutions in the third layer. The structure of our RBF neural networks is depicted in the following Figure 1.

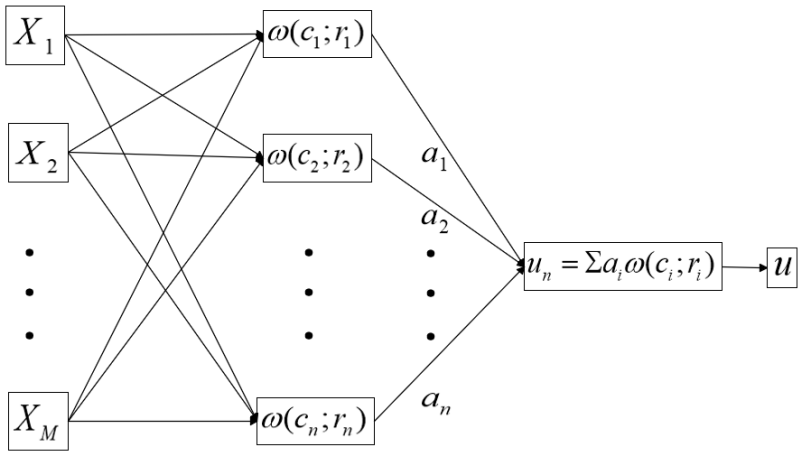

Figure 1. Architecture of RBF neural networks.

4. Numerical examples

In this section, we consider a series of numerical experiments with our proposed scheme to verify the higher accuracy and efficiency. Here, we directly recall some comments about the centers and shape parameters for two dimensional regions (for more details please refer to [22, 28]). Next, we consider the following Poisson equation with Dirichlet boundary condition:

$$\begin{cases} \Delta u(x,y) = f(x,y) \quad \text{in } \Omega, \\ u(x,y) = g(x,y) \quad \text{on } \partial\Omega, \end{cases} \tag{4.1}$$

where $f(x,y)$ and $g(x,y)$ are given functions, which can be derived from the exact solution, and the physical domain Ω is bounded with a Lipschitz boundary. Then, for (4.1), the corresponding entries of $\mathbf{F}$ on M collocation points in (3.3) become

$$\begin{cases} F_s = \Delta u(\mathbf{x}_s) - \sum\limits_{i=1}^{N} a_i \, \Delta\omega(c_i; r_i(\mathbf{x}_s)), \quad s = 1, \ldots, M_{\text{int}}, \\ F_t = \sqrt{\lambda} \left(u(\mathbf{x}_t) - \sum\limits_{j=1}^{N} a_j \, \omega(c_j; r_j(\mathbf{x}_t)) \right), \quad t = M_{\text{int}} + 1, \ldots, M. \end{cases} \tag{4.2}$$

Generally, the initial locations of the centers are set in the physical domain Ω. However, it has been observed that the centers in an extended region by a magnification factor $\eta > 1$ produce more accurate numerical results (more details please refer to [24]). For a physical domain Ω with center $\mathbf{x}_c$, the generalized centers are taken as

$$\tilde{\mathbf{x}}_i^0 = \eta(\mathbf{x}_i^0 - \mathbf{x}_c) + \mathbf{x}_c, \quad i = 1, \ldots, N, \tag{4.3}$$

which are placed over the generalized region. As mentioned in [22], the initial shape parameter values are crucial for the accuracy and efficiency of mesh-free methods. In this section, we select uniformly distributed initial values of the shape parameters in a given interval $[d_{\min}, d_{\max}]$, which are given by

$$c_i^{(0)} = d_{\min} + (d_{\max} - d_{\min})(i-1)/(N-1), \quad i = 1, \ldots, N, \tag{4.4}$$

and there are subsistent results for $d_{\max}$ and η, i.e., $2 < d_{\max} < 3$ and $2 < \eta < 3$ (see [22]).

In order to compare with the mesh-free methods listed in [22], we similarly employ a Halton distribution with $L = 300$ test points to calculate errors of u_N at the test points $\{\mathbf{x}_\ell\}_{\ell=1}^{L}$ in $\bar{\Omega}$. And to describe the accuracy of our proposed approximations, we recall the following two error indicators on the test data points:

$$e_r = \frac{||u - u_N||_{\infty,\bar{\Omega}}}{||u||_{\infty,\bar{\Omega}}}, \qquad e_s = \left(\frac{1}{L}\sum_{\ell=1}^{L}[u(\mathbf{x}_\ell) - u_N(\mathbf{x}_\ell)]^2\right)^{1/2}. \tag{4.5}$$

Considering various PDEs and domains, one can employ different RBF to efficiently approximate the continuous cases. To verify our numerical approach, we set the exact solution of (4.1) as

$$u(x, y) = -\frac{1}{2\pi^2}\sin(\pi x)\sin(\pi y), \quad \forall (x, y) \in \Omega. \tag{4.6}$$

In the following parts, we restrict (4.1) on unit square, peanut-shaped and five-petaled domains, respectively.

4.1. Square domain

For a unit square domain, we consider (4.1) with a homogeneous boundary condition (i.e., $g|_\Omega = 0$) and choose the Gaussian function (2.1) as the RBF. It is obvious that the collocation points have to be all located in the square, which guarantees that (4.1) exactly holds on the physical domain. However, the centers can be selected beyond the square to enhance the numerical accuracy.

In order to train the neural networks, by a distribution of Halton points, we choose 276 collocation points (200 points are selected in the interior and 76 points are uniformly distributed on the boundary). We also take a magnification factor $\eta = 2.2$ and the initial shape parameter values are evenly distributed within $[d_{\min}, d_{\max}] = [0.5, 2.5]$. The centers and collocation points of our extended region are shown in Figure 2, which are marked by red and blue, respectively.

We show the point-wise errors distributed on the physical domain in Figure 3 with a given $N_{\text{int}} = 16$ and magnitude scale 10^{-5}, which shows the global accuracy of our proposed schemes.

Meanwhile, we exhibit the convergence curves of e_s in Figure 4, which depict the accuracy of our numerical schemes with three different penalty factors in (3.4).

Notice that, to reduce the cost of calculations, one may set different locations of the centers. And one of the efficient choices is that number of the centers on the boundary (i.e., N_{bry}) is set as a half of it in the region (i.e., N_{int}). Hence, we choose different numbers of centers and collocation points to approximate (4.1) as follows.

Based on the above numerical results, one readily concludes that the orders of both e_r and e_s of our proposed schemes nearly increase by two orders when doubling the number of centers, and also achieves higher order (i.e., about one-order) than those in Section 4 in [22]. Meanwhile,

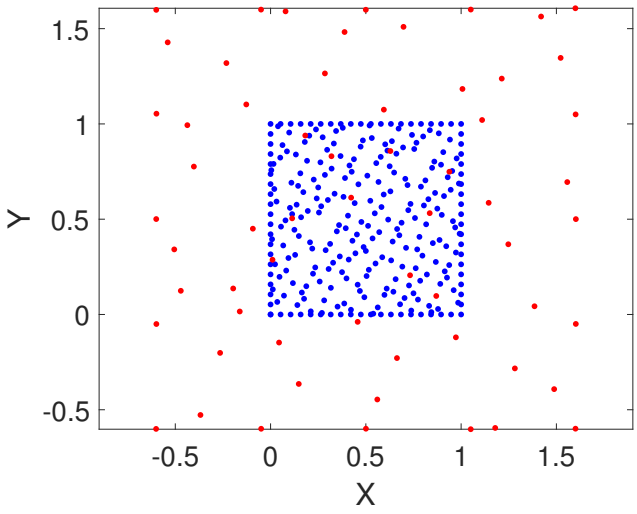

Figure 2. Centers(red) and collocation points (blue) of square.

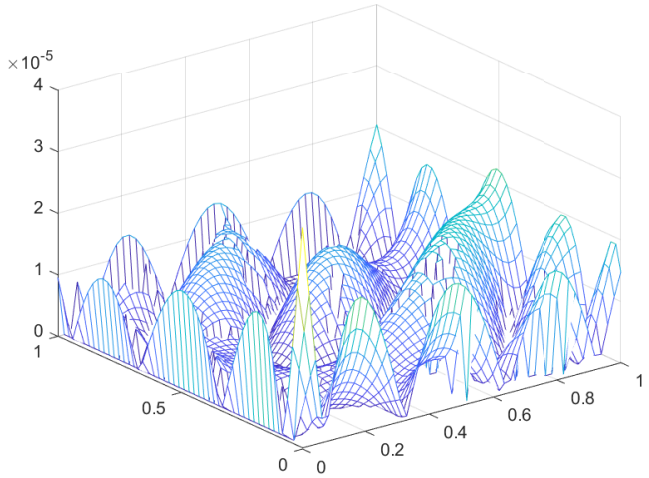

Figure 3. Point-wise errors on the square with $N_{\text{int}} = 16$ and $\lambda = 100$.

we can derive that the accuracy of our schemes decays quickly for different λ, which makes up the shortcoming of the approximations given in [22] for solving PDEs.

Specially, to state the benefit of trained shape parameters with different iterations, we choose suitable $d_{\min}$ and $d_{\max}$ to train the shape parameters according to (4.4). The corresponding numercial data are shown in the following Table 2, which indicates the numerical accuracy by e_r and e_s.

Additionally, we perform numerical tests to highlight the shortcomings of the prescribed shape parameters in Table 3. Although there is no theoretical analysis for reference, we only give numerical experiments and errors for verification.

The above data in Table 2 and Table 3 confirm that shape parameters with iteration training lead to substantially more precise approximations.

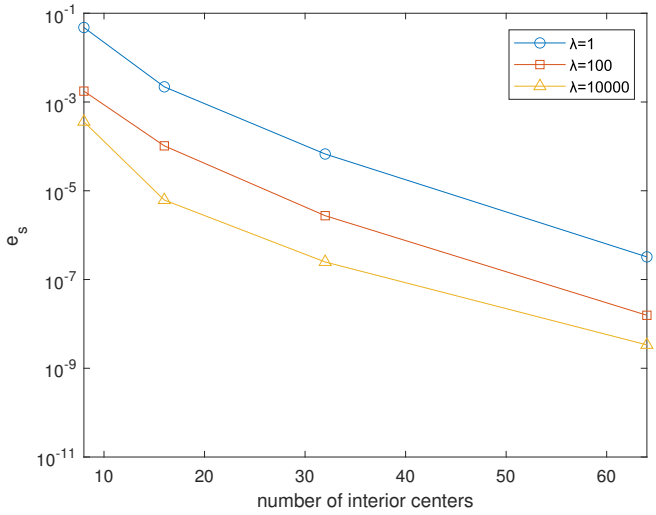

Figure 4. Convergence curves on the square.

Table 1. Error results of various N_{int} and N_{bry} with different λ.

	N_{int}	N_{bry}	e_r	e_s
$\lambda = 1$				
	16	8	1.279(-1)	2.197(-3)
	32	16	2.074(-3)	6.732(-5)
	64	32	1.776(-5)	3.237(-7)
$\lambda = 100$				
	16	8	7.133(-3)	1.028(-4)
	32	16	1.864(-4)	2.742(-6)
	64	32	1.221(-6)	1.563(-8)
$\lambda = 10000$				
	16	8	3.330(-4)	6.170(-6)
	32	16	2.012(-5)	2.483(-7)
	64	32	2.682(-7)	3.362(-9)

Table 2. Numerical data of different iterations with trained shape parameter.

$[c_{\min}, c_{\max}]$	$[d_{\min}, d_{\max}]$	iter	e_r	e_s
$[0.500, 2.477]$	$[0.5, 2.5]$	10	$2.493(-3)$	$5.409(-5)$
$[0.501, 2.446]$	$[0.5, 2.5]$	100	$6.013(-4)$	$7.538(-6)$
$[0.509, 2.885]$	$[0.5, 2.5]$	1000	$2.213(-4)$	$2.887(-6)$

Table 3. Numerical data of different iterations with constant shape parameters.

c_i	iter	e_r	e_s
0.5	10	2.705(−2)	4.768(−4)
0.5	100	1.857(−2)	3.211(−4)
0.5	1000	1.317(−2)	2.769(−4)

4.2. Peanut-shaped domain

Considering a smooth domain, we constraint (4.1) on a peanut-shaped region to check the efficiency of our numerical schemes. The corresponding boundary can be stated by

$$v(t) = \sqrt{\left(\frac{d}{dt}\,\mathrm{curve}(t)\right)^2 + \mathrm{curve}(t)^2}\ , \quad t \in (0, 2\pi)\ , \tag{4.7}$$

where

$$\mathrm{curve}(t) = 0.3\sqrt{\cos(2t) + \sqrt{1.1 - \sin^2(2t)}}\ .$$

For this case, we choose multiquadric functions to construct our numerical schemes. By $\eta = 2.2$ and $[d_{\min}, d_{\max}] = [0.1, 2.8]$, we show the centers and collocation points in an extended region with different colors in Figure 5. Obviously, all the collocation points are located on the peanut-shaped domain to guarantee the identities listed in (4.1).

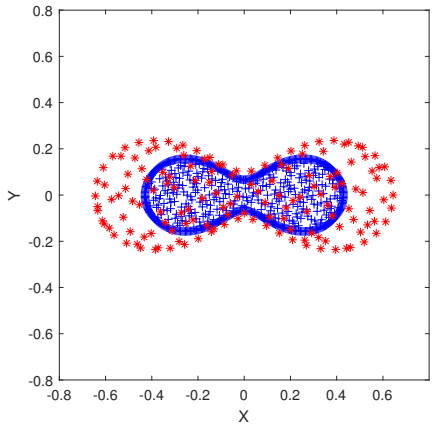

Figure 5. Centers(red) and collocation points (blue) of the peanut-shaped domain.

We give point-wise errors of numerical solutions on collocation points in the following Figure 6.

The corresponding magnitude scale 10^{-6} are depicted for $N_{\text{int}} = 20$ to show the high accuracy. Comparing with (4.1), the multiquadratic functions are also accurate for mesh-free methods to solve PDEs. Meanwhile, the curves of e_s with different λ are given in Figure 7, which illustrate the convergence rate of our numerical schemes.

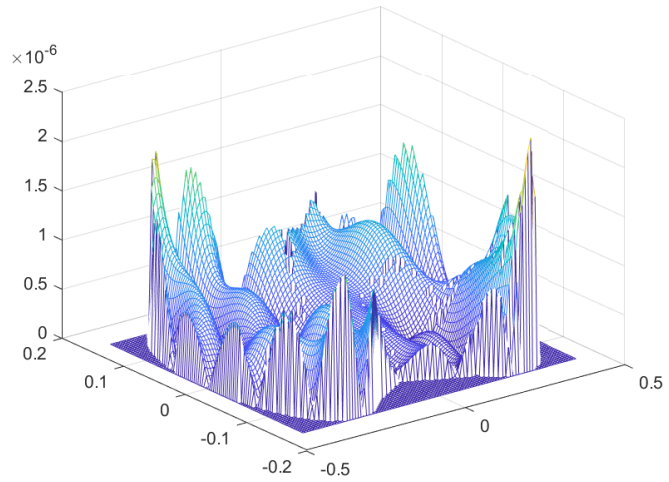

Figure 6. Point-wise errors on peanut-shaped domain with $N_{\text{int}} = 20$ and $\lambda = 100$.

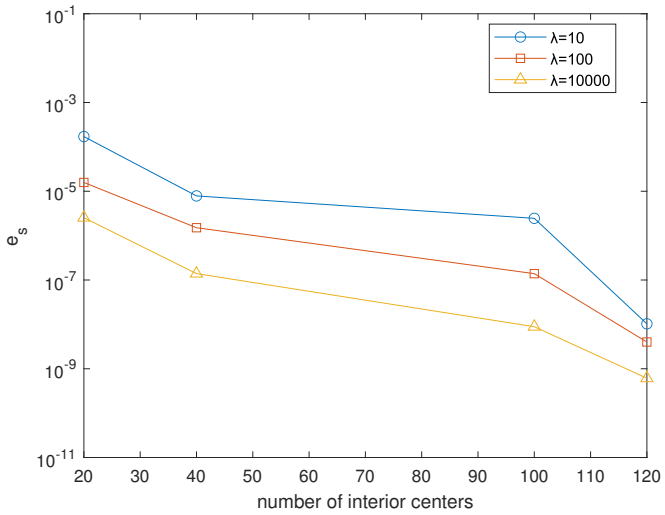

Figure 7. Convergence curves on peanut-shaped domain.

By the data shown in Table 4, comparing the same centers N (including N_{int} and N_{bry}, respectively), it can be concluded that increasing the number of centers can improve the accuracy rapidly. Meanwhile the penalty factor λ also directly affects the accuracy of the numerical solution.

4.3. Five-petaled domain

A more complex geometric region, five-petaled region, is selected to illustrate the efficiency of our proposed scheme. The boundary $\partial\Omega$ of the five-petaled region is described by the polar coordinate:

$$r(\theta) = 1 + \cos^2(5\theta/2), \quad 0 \le \theta \le 2\pi. \tag{4.8}$$

Table 4. Error results of various N_{int} and N_{bry} with different λ.

	N_{int}	N_{bry}	e_r	e_s
$\lambda = 10$				
	20	10	3.403(-2)	1.707(-4)
	40	20	1.512(-3)	7.852(-6)
	100	50	3.864(-4)	2.453(-6)
$\lambda = 100$				
	20	10	2.766(-3)	1.578(-5)
	40	20	2.948(-4)	1.507(-6)
	100	50	2.711(-5)	1.390(-7)
$\lambda = 10000$				
	20	10	4.956(-4)	2.542(-6)
	40	20	3.869(-5)	1.394(-7)
	100	50	1.674(-6)	8.865(-9)

We also choose the Gaussian function (2.1) to discretize the models. In Figure 8, we present the locations of the centers and collocation points with $\eta = 2.8$ and $[d_{\min}, d_{\max}] = [0.5, 2.9]$.

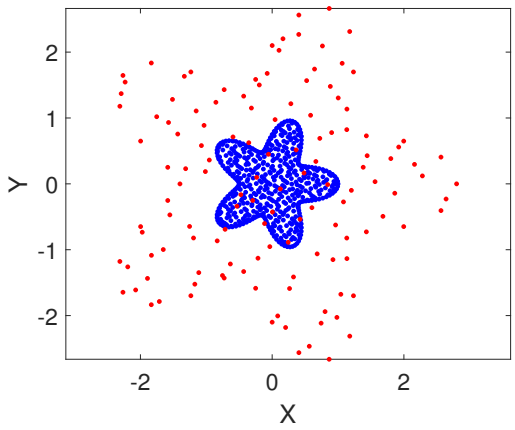

Figure 8. Centers(red) and collocation points (blue) of five-petaled region.

The point-wise errors of our numerical solutions with $N_{\text{int}} = 40$ and magnitude scale 10^{-7} are depicted in Figure 9. Similarly, by setting different N, we readily show the global accuracy of our proposed schemes for (4.1).

In the following Figure 10, we give the tendency of e_s as the numbers of centers increase with three different λ.

Similarly, as mentioned earlier, we set the centers on the boundary to be half of the interior centers. To illustrate the approximation effect of our proposed numerical schemes, we choose different numbers of centers and collocation points to approximate (4.1), and list the corresponding numerical results as follows.

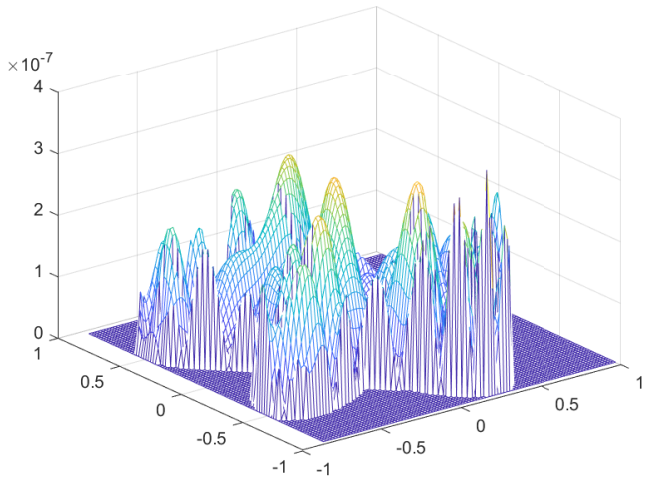

Figure 9. Point-wise errors on five-petaled domain with $N_{\text{int}} = 40$ and $\lambda = 100$.

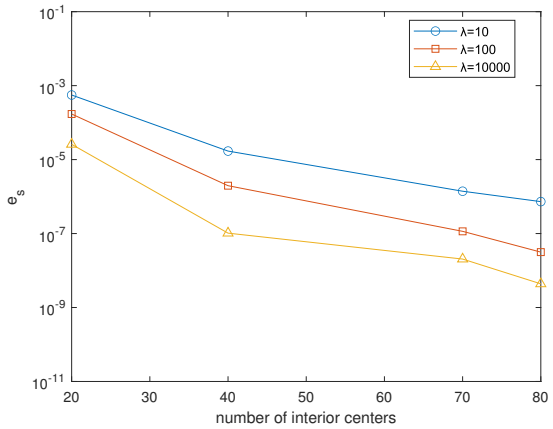

Figure 10. Convergence curves on five-petaled domain.

With the help of above data, we find that the errors achieve one higher order for $\lambda = 10,\ 100,\ 10000$ with the same N_{int} and N_{bry}, respectively. It verifies the fact that the accuracy of approximations on the boundary directly affects the global accuracy on the domain, which is entirely consistent with the classic result.

4.4. Results comparison

Now, we are going to compare our numerical scheme with the given method in [22] and Kansa's method in [21], respectively.

For simplicity, focusing on a five-petaled domain with the same $[d_{\min}, d_{\max}] = [0.5, 2.9]$ and $\eta = 2.8$, we recall the data listed in Table 4 of [22] to illustrate the precision of our proposed schemes.

By the above data, considering the same size of N_{int} and N_{bry}, the numerical results obtained by our scheme are about one-order and two-order higher than the method given in [22] and

Table 5. Error results of various N_{int} and N_{bry} with three different λ.

	N_{int}	N_{bry}	e_r	e_s
$\lambda = 10$				
	20	10	5.116(-2)	5.576(-4)
	40	20	1.594(-3)	1.700(-5)
	80	40	5.160(-5)	7.302(-7)
$\lambda = 100$				
	20	10	9.675(-3)	1.709(-4)
	40	20	2.504(-4)	1.969(-6)
	80	40	2.200(-6)	8.616(-8)
$\lambda = 10000$				
	20	10	1.642(-3)	2.590(-5)
	40	20	6.167(-5)	1.026(-7)
	80	40	5.053(-7)	4.369(-9)

Table 6. Comparisons of the current three methods.

N_{int}	N_{bry}	Our scheme		Method in [22]		Kansa's method	
		e_r	e_s	e_r	e_s	e_r	e_s
40	40	3.155(-5)	4.862(-7)	5.512(-5)	9.011(-7)	6.185(-5)	4.940(-7)
50	40	1.126(-5)	2.378(-7)	1.322(-5)	3.017(-7)	8.795(-6)	6.113(-8)
80	50	2.075(-7)	1.067(-9)	5.334(-8)	1.120(-9)	3.946(-6)	4.484(-8)
90	40	2.908(-8)	4.371(-10)	5.765(-8)	8.892(-10)	3.291(-6)	2.771(-8)
100	50	2.796(-8)	4.063(-10)	4.260(-8)	8.160(-10)	1.283(-6)	2.328(-8)

traditional Kansa's method, respectively.

In order to further demonstrate the accuracy and efficiency of our scheme, we choose some numerical solutions with different N_{int} and plot the error curves to depict the convergence behaviors of the approximate solutions in the following figure.

Especially, the error indicator of our numerical method arrives 10^{-10} quickly, but the method in [22] can nearly reach 10^{-9} and Kansa's method reads 10^{-8}, which shows better convergence behaviours of our numerical scheme than the other two numerical methods. Furthermore, the results listed in Table 6 also allow us to state that the accuracy of our scheme improves rapidly with the increasing numbers of center points, which indicates the efficiency of our proposed RBF neural networks schemes for solving the governed PDEs models with some boundary conditions.

5. Conclusions

In this paper, we propose a novel method based on RBF neural networks to solve second-order PDEs on different kinds of domains. Specially, we employ fixed centers to construct

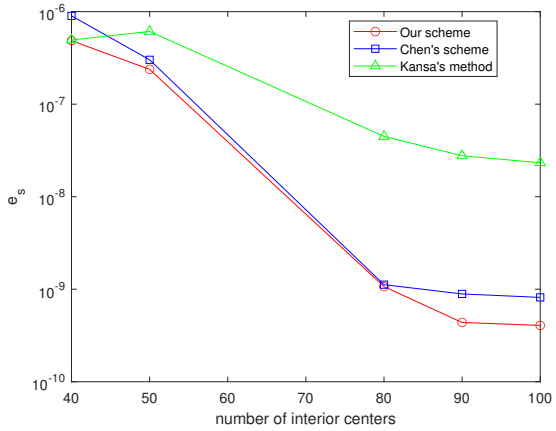

Figure 11. Convergence curves of three schemes.

the improved RBF neural networks schemes for solving the minimal functions by nonlinear least squares. Our proposed schemes avoid updating the points within per iterations and save computational storage, which is distinct from current approximations shown in [22]. Comparing with the RBF neural networks method in [22], our numerical scheme does not require calculating derivations of the exact Jacobian of the corresponding nonlinear system, which will save lots of memory within either trainings or iterations. Based on the result that extending the centers' locations to the outside of physical domain will improve the accuracy of numerical calculations, we extend the calculating domain to a new region by some magnification factors. In the numerical experiments, we use both Gaussian and multiquadratic functions to approximate PDEs in three different domains. Meanwhile, we fix the centers by a distribution of Halton points to reduce the complexity. And the numerical results show that by increasing the number of center points, the accuracy will also be rapidly enhanced. In the case of a larger number of center points in the same domain, we obtain that the accuracy of our scheme is about one order higher than the above related numerical methods. In the coming work, we will investigate novel techniques for improving the numerical errors to machine accuracy and extend our scheme to solve generalized PDEs using other kinds of RBF, including nonlinear, high order and evolutionary PDEs on complex regions.

References

[1] M. Abbaszadeh, A. R. Salec and A. H. Taghreed, *A radial basis function (RBF)-finite difference method for solving improved Boussinesq model with error estimation and description of solitary waves*, Numerical Methods for Partial Differential Equations, 2024, 40, e23077.

[2] O. Akbilgic, H. Bozdogan and M. E. Balaban, *A novel hybrid RBF neural networks model as a forecaster*, Statistics and Computing, 2014, 24, 365–375.

[3] M. M. Alqezweeni, R. A. Glumskov, V. I. Gorbachenko, et al., *Solving partial differential equations on radial basis functions networks and on fully connected deep neural networks*, Proceedings of the International Conference on Intelligent Vision and Computing, 2022, 15, 240–249.

[4] M. M. Alqezweeni, V. I. Gorbachenko, M. V. Zhukov, et al., *Efficient solving of boundary*

value problems using radial basis function networks learned by trust region method, International Journal of Mathematics and Mathematical Sciences, 2018, 1, 9457578.

[5] A. S. Bandeira, K. Scheinberg and L. N. Vicente, *Convergence of trust-region methods based on probabilistic models*, SIAM Journal on Optimization, 2014, 24(3), 1238–1264.

[6] C. S. Chen, A. Karageorghis and F. F. Dou, *A novel RBF collocation method using fictitious centres*, Applied Mathematics Letters, 2020, 101, 106069.

[7] W. Chen and Z. J. Fu, *Recent Advances in Radial Basis Function Collocation Methods*, Springer, 2013.

[8] M. Dehghan and A. Shokri, *A numerical method for two-dimensional Schrödinger equation using collocation and radial basis functions*, Computers & Mathematics with Applications, 2007, 54, 136–146.

[9] F. F. Dou and Y. C. Hon, *Fundamental kernel-based method for backward space-time fractional diffusion problem*, Computers & Mathematics with Applications, 2016, 71, 356–367.

[10] L. N. Elisov, V. I. Gorbachenko and M. V. Zhukov, *Learning radial basis function networks with the trust region method for boundary problems*, Automation and Remote Control, 2018, 79, 1621–1629.

[11] G. Fairweather and A. Karageorghis, *The method of fundamental solutions for elliptic boundary value problems*, Advances in Computational Mathematics, 1998, 9, 69–95.

[12] M. Farhan, Z. Omar, F. M. Oudina, et al., *Implementation of the one-step one-hybrid block method on the nonlinear equation of a circular sector oscillator*, Computational Mathematics and Modeling, 2020, 31, 116–132.

[13] J. T. Fei and H. F. Ding, *Adaptive sliding mode control of dynamic system using RBF neural network*, Nonlinear Dynamics, 2012, 70, 1563–1573.

[14] G. Garmanjani, M. Esmaeilbeigi and R. Cavoretto, *Adaptive residual refinement in an RBF finite difference scheme for 2D time-dependent problems*, Computational & Applied Mathematics, 2024, 43, 39.

[15] M. A. Golberg and C. S. Chen, *The method of fundamental solutions for potential, Helmholtz and diffusion problems*, in: M. A. Golberg (Ed.), *Boundary integral methods: Numerical and mathematical aspects*, WIT Press/Comput. Mech. Publ., Boston, MA, 1999, 1, 103–176.

[16] V. I. Gorbachenko and M. V. Zhukov, *Solving boundary value problems of mathematical physics using radial basis function networks*, Computational Mathematics and Mathematical Physics, 2017, 57, 145–155.

[17] M. Haghi, M. Ilati and M. Dehghan, *A radial basis function-Hermite finite difference (RBF-HFD) method for the cubic-quintic complex Ginzburg–Landau equation*, Computational and Applied Mathematics, 2023, 42(3), 115.

[18] M. A. Jankowska and A. Karageorghis, *Variable shape parameter Kansa RBF method for the solution of nonlinear boundary value problems*, Engineering Analysis with Boundary Elements, 2019, 103, 32–40.

[19] M. A. Jankowska, A. Karageorghis and C. S. Chen, *Kansa–RBF algorithms for elliptic BVPs in annular domains with mixed boundary conditions*, Mathematics and Computers in Simulation, 2023, 206, 77–104.

[20] M. A. Jankowska, A. Karageorghis and C. S. Chen, *Training RBF neural networks for solving nonlinear and inverse boundary value problems*, Computers & Mathematics with Applications, 2024, 165, 205–216.

[21] E. J. Kansa, *Multiquadrics-a scattered data approximation scheme with applications to computational fluid dynamics. II. Solutions to parabolic, hyperbolic and elliptic partial differential equations*, Computers & Mathematics with Applications, 1990, 19, 147–161.

[22] A. Karageorghis and C. S. Chen, *Training RBF neural networks for the solution of elliptic boundary value problems*, Computers & Mathematics with Applications, 2022, 126, 196–211.

[23] E. Lehto, V. Shankar and G. B. Wright, *A radial basis function (RBF) compact finite difference (FD) scheme for reaction-diffusion equations on surfaces*, Society of Industrial and Applied, 2017, 39(5), 2129–2151.

[24] L. Ling, R. Opfer and R. Schaback, *Results on meshless collocation techniques*, Engineering Analysis with Boundary Elements, 2006, 30, 247–253.

[25] G. R. Liu, *Mesh Free Methods: Moving Beyond the Finite Element Method*, CRC Press, Boca Raton, 2002.

[26] P. Priyanka, S. Arora and F. M. Oudina, et al., *Superconvergence analysis of fully discrete Hermite splines to simulate wave behavior of Kuramoto–Shivashinsky equation*, Wave Motion, 2023, 121, 103187.

[27] P. Priyanka, F. M. Oudina, S. Sahani, et al., *Travelling wave solution of fourth order reaction diffusion equation using hybrid quintic Hermite splines collocation technique*, Arabian Journal of Mathematics, 2024, 13, 341–367.

[28] S. A. Sarra and D. Sturgill, *A random variable shape parameter strategy for radial basis function approximation methods*, Engineering Analysis with Boundary Elements, 2009, 33, 1239–1245.

[29] C. Satyanarayana, M. K. Yadav and M. Nath, *Multiquadric based RBF-HFD approximation formulas and convergence properties*, Engineering Analysis with Boundary Elements, 2024, 160, 234–257.

[30] T. Tanbay and B. Ozgener, *A comparison of the meshless RBF collocation method with finite element and boundary element methods in neutron diffusion calculations*, Engineering Analysis with Boundary Elements, 2014, 46, 30–40.

Received February 2025; Accepted January 2026; Available online February 2026.